BINGO!: A Novel Pruning Mechanism to Reduce the Size of Neural Networks

Aditya Panangat, Flower Mound High School

ABSTRACT

Over the past decade, the use of machine learning has increased exponentially. Models are far more complex than ever before, growing to gargantuan sizes and housing millions of neurons. Unfortunately, the fact that large models have become the state of the art means that it often costs millions of dollars to train and operate them. These expenses not only hurt companies but also bar non-wealthy individuals from contributing to new developments and force consumers to pay greater prices for AI. Moreover, each time models are trained, because of their sheer size, tons of CO2 are emitted into the atmosphere, fueling global warming. Current methods used to prune models, such as iterative magnitude pruning, have shown great accuracy but require an iterative training sequence that is incredibly computationally taxing. To solve this problem, BINGO is introduced. BINGO, during the training pass, studies specific subsets of a neural network one at a time to gauge how significant of a role each neuron plays in providing a network its accuracy. By the time training is done, BINGO will generate a significant score for each neuron, allowing for insignificant neurons to be pruned in one shot. The goal of BINGO is to provide a pruning technique that is less computationally intensive than current pruning techniques which doesn't decrease model accuracy, allowing for a world where AI growth doesn't have to mean model growth, as well.

MOTIVATION AND APPROACH

Machine learning developers like to go big, training models with millions of neurons. In fact, according to Howarth (2024), OpenAI's GPT-4 has 1.8 trillion parameters. Models are

expected to be large to meet performance and accuracy standards; however, as these models grow, a few key issues arise.

First, training becomes incredibly expensive. Training billions of neurons means performing forward passes to compute activations, backward passes to calculate gradients for all associated parameters, and updating these parameters iteratively across potentially millions of data samples, which is incredibly computationally expensive. According to Li (2020), it costs OpenAI about 4.6 million dollars to train their GPT-3 model just once. Moreover, according to Slowik (2023) because of the sheer size of Open AI's models, they can cost up to 1.5 million dollars per year for operational usage.

Second, this is a great barrier to accessibility. New developers are unable to afford these computational costs, preventing them from industry entry. The businesses that can afford this expense pass this cost down to consumers, increasing the price of machine learning models for clients. This is detrimental when it comes to buying machine learning models essential to crucial practices, such as illness detection or education.

Third, copious amounts of energy is used, hurting the environment via emissions. According to Hao (2019) training a model just once can emit more than 626,000 pounds of CO2, with Patterson et. al (2021) finding that GPT-3 causes the emission of 502 metric tons of CO2 per training session.

Table 1 created by Strubell et al. (2020), showing the energy efficiency of training models. CO2e estimates carbon dioxide emissions (lbs) produced by training a model once. Cloud computing cost resembles the cost (USD) of training a model once.

Model	Hardware	Power (W)	Hours	kWh·PUE	CO_2e	Cloud compute cost
$\text{Transformer}_{base}$	P100x8	1415.78	12	27	26	$41–$140
Transformer_{big}	P100x8	1515.43	84	201	192	$289–$981
ELMo	P100x3	517.66	336	275	262	$433–$1472
BERT_{base}	V100x64	12,041.51	79	1507	1438	$3751–$12,571
BERT_{base}	TPUv2x16	—	96	—	—	$2074–$6912
NAS	P100x8	1515.43	274,120	656,347	626,155	$942,973–$3,201,722
NAS	TPUv2x1	—	32,623	—	—	$44,055–$146,848
GPT-2	TPUv3x32	—	168	—	—	$12,902–$43,008

Frankle and Carbin (2018) at MIT CSAIL articulate a hypothesis dubbed *the lottery ticket hypothesis*: they substantiated that dense, randomly-initialized, feed-forward networks contain subnetworks (called winning tickets) that—when trained in isolation—reach test accuracy comparable to the original network in a similar number of iterations. Essentially, they argue that within every dense, trained neural network, there lies certain "winning tickets"–smaller networks that can achieve the same accuracy as the full model. Thus, solving the issues that arise from the existence of enormous models boils down to one question: how can these smaller models–these winning tickets–be found?

The most cost-effective approach that has been explored is called neural network pruning. Pruning relies on finding neurons in a network that do not contribute significantly to the model's accuracy. In this approach, after training is complete, neurons with weights closest to zero are assumed to be insignificant and are removed. However, Rad and Seuffert (2024) show that this technique of one-shot pruning has been shown to drastically reduce accuracy at higher pruning rates. Even pruning just 10% of a network this way has been shown to result in accuracy dropping by 30 percentage points. Although pruning is efficient, it sacrifices accuracy.

A well-performing pruning method has been found by MIT researchers Frankle and Carbin (2018). They propose a nuanced version of *iterative magnitude pruning* (IMP). IMP is the best-performing pruning method so far established, and it functions like so:

A. A dense neural network is initialized, with each neuron holding a randomly generated number as its weight. These initial weights are stored. This network is then trained to completion.
B. Then, this 3-step process is performed iteratively:

1. Neurons with weights close to zero are pruned (removed) from the network, creating a smaller, pruned structure for the network
2. The weight of every neuron in this new, pruned structure is reset to its initial assignment weight (saved in step A)
3. This pruned structure is trained once again until it achieves accuracy similar to the original model

The process of setting the weights of the pruned architecture back to their initial weights and retraining allows for the unimportant neurons to be identified repeatedly as the model is pruned slowly. On the MNIST dataset, Frankle and Carbin (2018) found that this version of IMP was able to remove 96% percent of the network's neurons while maintaining the same test accuracy. However, the issue with this approach is that the neural network must be trained and retrained iteratively in order to prune. Training a model just once is costly (4.6 million dollars to train GPT-3), and hence IMP is too expensive.

PROJECT LOGISTICS AND ORGANIZATION

Although IMP works extraordinarily well, its major pitfall is the fact that it requires multiple training sessions in order to complete pruning. IMP, as a response to the lottery ticket hypothesis, acts in a way similar to the lottery: all the lottery numbers are printed, and only then a winner is found.

The ideal technique to reduce the size of models then is one that doesn't require additional training sessions to be done. So, I introduce BINGO, a novel neural network pruning technique that addresses the lottery ticket hypothesis not by iteratively finding unnecessary neurons through additional training sessions, but by identifying which neurons will be unnecessary while the original model itself is training. Rather than finding winning tickets after

all the winning numbers are called out, people find BINGO one letter at a time while the game is taking place. Simply put, BINGO solves the issue of needing additional iterative training sessions for model pruning by collecting information during the original training session. With this information, after the model is trained, all unimportant neurons shall be pruned from the model in one shot. This means that computationally expensive, repetitive training and resetting of the model is not necessary, drastically reducing the computational and environmental cost required to run fast, efficient models. When models are faster for cheaper, machine learning is a less environmentally taxing and more accessible field.

BINGO achieves computationally efficient pruning by contributing 2 novel, creative, and effective mechanisms to the literature.

First, I propose a technique to isolate possible lottery tickets dubbed *ticket searching*. This technique borrows from neural network dropout, a regularization technique proposed by Srivastava et al. (2014), which attempts to make models more generalizable by deactivating a random subset of neurons during each training pass. In a similar way, ticket searching, just after each training pass, sets a subset of neurons back to their *initial weights*. It is important to know that this process does not actually affect the weights in the network. Instead, just after each real training pass, this technique simply simulates a "fake training pass," setting some neurons back to their initial assignments to simply keep score of what *would have* happened if those were the true neuron weights.

Second, I propose a new measurement called *neuron significance*. Based on the way that the accuracy of a model is calculated to change during each ticket search, BINGO will calculate a neuron significance metric for each neuron in the neural network. Neuron significance for each neuron will be scaled from 0 to 1, with higher scores representing neurons that are believed to

contribute significantly to model accuracy. This metric will be calculated using data that is gathered from ticket searching, which will keep track of what happens to accuracy when certain subsets of neurons are set back to their initial, untrained values. Data from each ticket search will be used to calculate the total significance score for each neuron. Currently, I have created a calculation for significance (S_n) which is computed just after each ticket search:

$\mathbf{S_n = 1 - (|A_{before} - A_{after}|) / A_{before}}$ (Where A_{before} is the model accuracy before resetting the neuron; A_{after} is the model accuracy after resetting the neuron)

Finally, based on neuron significance scores, BINGO will prune neurons until a certain minimum accuracy is reached, pruning neurons of lowest significance first.

BINGO is superior to existing solutions. First, because it functions much faster than any accurate solution in the state of the art. This is because all BINGO requires is an extra (without gradient update) ticket search process after training. Unlike IMP, BINGO does not require iterative training and pruning to occur. BINGO improves upon current technology. Namely, *ticket searching* is inspired by dropout, but instead of dropping neurons, it resets the values back to their initial assignments. I chose to reset back to initial weights after experimenting with both methods. When significance scores were calculated based on data where: a) Random neurons were dropped during each ticket search and b) random neurons were reset back to their initial weights during each ticket search, I found that the pruned models were 50% more accurate in option b). This means the significance scores were more accurate in predicting truly significant neurons when neurons were reset back to initial weights rather than dropped.

On MNIST, IMP does perform better than BINGO. IMP was able to remove 96% of neurons while maintaining accuracy, while my rough-draft version of BINGO was only able to remove about 20%. However, this is largely attributable to the fact that my calculation for

significance score is very rudimentary. Regardless, BINGO was able to remove 20% of the neurons on an MNIST-trained network without the iterative retraining that IMP uses, which is very promising.

There are 3 steps required to implement my project proposal:

1. Develop *ticket searching*, a dropout-esque mechanism that switches a subset of neurons to their initial assignments
2. Experiment with more targeted ticket searching. Find a ticket searching mechanism that will choose which subset of neurons to drop strategically by slowly focusing on finding truly unproductive subsets of neurons.
3. Develop a significance score calculation that shows truly how significant each neuron is in a network.

This project is technically feasible. Computational costs will never be extraordinarily high because I will be able to build BINGO by testing on smaller models first.

First and foremost, BINGO is faster than IMP. I have a rough draft version of BINGO that is able to finish pruning an MNIST-trained model in 8 minutes and 47 seconds (including training time). On the other hand, running IMP on my machine as proposed by Frankle and Carbin (2018) finishes pruning in 13 minutes and 5 seconds.

Second, BINGO and IMP tend to find similar neurons to be insignificant. Empirically, when I ran both BINGO and IMP on an MNIST-trained neural network, 45% of the neurons they removed 10 epochs in were the same. Once a better significance score calculation is determined, I predict that BINGO and IMP will be removing the same neurons a higher percentage of the time.

References

Frankle, J., & Carbin, M. (2018). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. https://arxiv.org/pdf/1803.03635

Hao, K. (2019, June 6). *Training a single AI model can emit as much carbon as five cars in their lifetimes*. MIT Technology Review; MIT. https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/

Howarth, J. (2024, August 6). *Number of Parameters in GPT-4 (Latest Data)*. Exploding Topics; Exploding Topics. https://explodingtopics.com/blog/gpt-parameters

Li, C. (2020, June 3). *OpenAI's GPT-3 Language Model: A Technical Overview*. Lambdalabs.com; Lambda, Inc. https://lambdalabs.com/blog/demystifying-gpt-3?srsltid=AfmBOopEBwnjFUkDPIihPXftrT59qpn_ODdaWJp7VNXC83-PRqT1xYMH

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., & Dean, J. (2021). *Carbon Emissions and Large Neural Network Training*. https://arxiv.org/ftp/arxiv/papers/2104/2104.10350.pdf

Rad, J., & Seuffert, F. (2024, September 27). *Investigating the Effect of Network Pruning on Performance and Interpretability*. Arxiv.org. https://arxiv.org/html/2409.19727v1

Slowik, C. (2023, February 16). *How Much Does It Cost to Use GPT? GPT-3 Pricing Explained*. Neoteric Custom Software Development Company. https://neoteric.eu/blog/how-much-does-it-cost-to-use-gpt-models-gpt-3-pricing-explained/

Srivastava, N., Hinton, G., Krizhevsky, A., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958. https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Strubell, E., Ganesh, A., & McCallum, A. (2020). Energy and Policy Considerations for Modern Deep Learning Research. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(09), 13693–13696. https://doi.org/10.1609/aaai.v34i09.7123

White, C., Ai, Safari, M., Sukthanker, R., Elsken, T., Hutter, F., 2022, C., White, M., Safari, R., Sukthanker, B., Ru, T., Elsken, A., Zela, D., Dey, F., & White, H. (2023). *Neural Architecture Search: Insights from 1000 Papers*. https://arxiv.org/pdf/2301.08727

Yu, R., Li, A., Chen, C.-F., Lai, J.-H., Vlad, Morariu, I., Han, X., Gao, M., Lin, C.-Y., Larry, & Davis, S. (2017). *NISP: Pruning Networks using Neuron Importance Score Propagation*. https://openaccess.thecvf.com/content_cvpr_2018/papers/Yu_NISP_Pruning_Networks_CVPR_2018_paper.pdf